Date
Page No.

Assignment

Name - Jyoti Section - CE Roll no. - 13

Q2

```
for(i=1 to n)
{ i = i*2 }
```

1 2 4 8 ... n

$2^{k-1} = n$

$(k-1)\log 2 = \log n$

$k = \log_2 n + 1$

Time complexity = $O(\log_2 n)$

Q9 $T(n) = 3T(n-1)$, $n > 0$

$T(n-1) = 3T((n-1)-1)$

$T(n-1) = 3T(n-2)$

$T(n) = (3T(n-2))$

$T(n) = 9T(n-2)$

$T(n-2) = 3T(n-3)$

$T(n) = 27T(n-3)$

$T(n) = 3^k T(n-k)$

$n - k = 0$

$n = k$

$T(n) = 3^n * 1$

$T(n) = 3^n$ Ans

Time complexity = $O(3^n)$

DELTA Notebook

Date
Page No.

Q4 $T(n) = 2T(n-1)$

$T(n-1) = 2T(n-2)$

$T(n) = 4T(n-2)$

$T(n-2) = 2T(n-3)$

$T(n) = 8T(n-3)$

$T(n) = 2^k T(n-k)$

$n-k=0$

$T(n) = 2^n$. Ans

Time complexity = $O(2^n)$

Q5 $i=1$

$s=1$

The value of i increases by 1 at each interval. $n = \frac{k(k+1)}{2} \Rightarrow k = \sqrt{n}$

$T(n) = O(\sqrt{n})$

Q6 $i^2 <= n \Rightarrow i <= \sqrt{n}$

Time comp = $O(\sqrt{n})$ Ans

Q7 For Ist loop → $n/2$

For IInd loop → $\log n$

For IIIrd loop → $\log n$

Time complexity = $O(n(\log n)^2)$.

Date
Page No.

Q8 n^2 → Recurrence.

$$T(n) = T(n-3)$$
$$T(n-3) = T(n-6)$$
$$T(n) = T(n-6)$$
$$T(n) = T(n-9)$$
$$T(n) = T(n-3^k)$$

$$n - 3^k = 0$$
$$n = 3^k$$
$$\log n = k \log 3$$
$$k = \log_n 3$$

OR $n^2 > \log_n 3$

Time complexity = $O(n^3)$

Q9 $n + n/2 + n/3 + n/4 + \cdots n/n$

$$n\left(1 + 1/2 + 1/3 + 1/4 + \cdots + \frac{1}{n}\right)$$

$(\log n)$

Time complexity = $n \log n$

Date
Page No.

Q 11)

```
void fun (int n)
{
    int j=1, i=0;
    while (i<n)
    {
        i = i+j;
        j++;
    }
}
```

1
1+2
1+2+4
1+2+4+5

last term will be

$\frac{k(k+1)}{2} = n$

$k^2 = n$

$k = \sqrt{n}$

Time complexity = $O(\sqrt{n})$

Q14 :-

$T(n) = T(n/4) + T(n/2) + cn^2$

Assuming $T(n/2) > T(n/4)$

$T(n) = T(n/2) + cn^2$

Applying master's theorem.

$a = 1 \quad b = 2$

$f(n) = cn^2 \mid n^{\log_2 2}$

Date
Page No.

As, $cn^2 > n^{\log 2}$

So, $T(n) = O(f(n))$

$T(n) = n^2$ Ans

$= O(n^2)$

Q15 $n + n/2 + n/3 + n/4 + \cdots$

$n(1 + 1/2 + 1/3 + 1/4 + \cdots)$

$n(\log n) = n \log n$ Ans

Q 16

$n = i^k$

$\log n = k \log i$

$k = \log_i n$

$O(\log n)$ Ans

Q 18 :-

(a) $100 < \log\log n < \log n < \log(n!) < \text{root } n < n < n \log n < n^2 < 2^n < 2^{2n} = 4^n < n!$

(b) $1 < \log\log(n) < \sqrt{\log(n)} < \log(n) < \log 2n < 2\log n < n < 2n < 4n < n \log n < \log(n!) < n^2 < 2(2^n) < n!$

(c) $96 < \log_8 n < \log_2 n < \log(n!) < 5n < n \log n < n \log_2 n < 8n^2 < 7n^3 < 8^{2n-6} < n!$

DELTA Notebook

Date
Page No.

Q1 Assy

Assymptotic notation are languages that allow us to analyze an algorithm runs time delay identifying its behaviour as the input size of algorithms.

Types :-

(i) Big O :- It is commonly used for worst case and give us upper bond for the growth rate of run time of algorithm
Eg = O(n)

(ii) Big Omega :- Provides asymptotic lower bond.

(iii) Theta :- Tight bond of the growth rate of runtime of algo.

Q7 Small :- It is used to denote the upper bond

Q20 Insertion Sort :-

Date
Page No.

(i) Insertion Sort by iterative way :-

```
Void Insertion Sort (int arr[], int n)
{
    int i, temp, j ;

    for (i ← 1 to n)
    {
        temp ← arr[i]
        j ← i-1;
        while (j >= 0 AND arr[j] > temp)
        {
            arr[j+1] ← arr[j];
            j ← j-1;
        }
        arr[j+1] ← temp;
    }
}
```

(ii) ~~By Recu~~

Date
Page No.

(ii) By Recursive :-

```
insertion (int a[], int i, int n)
{
    int val = a[i];
    int j = i-1;

    while (j>0 && a[j-1] > value)
    {
        a[j] = a[j-1];
        j--;
    }
    a[j] = val;
    if (i+1 <= n)
        insertion (a, i+1, n);
}
```

Qn :- Since polynomials grow slower than exponentially n^k has an asymptotic upper bond of $O(a^n)$.